University of Southern Denmark

DM852

Heuristics & Approximations Algorithms

Submitted To:
Marco Chiarandini
Lene Monrad Favrholdt
IMADA
Mathematics & Computer Science Department

Submitted By :
Alexander Lerche Falk
Narongrit Unwerawattana
Spring - Master of Computer Science

Contents

1 Introduction

This project shows metaheuristic algorithms, resolving the Capacitated Vehicle Routing Problem (CVRP). The difference between metaheuristics and heuristics is in the solution part. For heuristics, you are trying your best to find a solution, even though it is not optimal. The algorithm is adapted to the problem in such a greedy approach, it can get stuck in a local optimum. This is fine since it is the idea of heuristics: "just solve the problem".
Metaheuristics are less greedy and tends to be more problem independent. They accept temporary solutions and allow "bad" steps as an attempt to get a better solution. You have a local- and global optimum to keep track of the best solution found. You can say metaheuristics are exploting heuristics to avoid getting trapped. In this metaheuristics implementation project, we have chosen to implement two algorithms for CVRP: Simulated Annealing (SA) and Ant Colony Optimization (ACO). The SA algorithm is the one we have uploaded for electronic submisison at http://valkyria.imada.sdu.dk/D0App/. The ACO algorithm is implemented but does not perform well. We will compare the algorithms with our previous heuristic / local search project and lastly, show our results of computation for our two algorithms.

— Alexander & Narongrit

2 Simulated Annealing

The Simulated Annealing (SA) algorithm is inspired from the annealing process in metal, where you are altering the physical state of the metal by heating and cooling it. The inspiration can be used in computer science as well. We can use the algorithm on CVRP by starting off by generating a solution to the problem and not "care" about the initial solutions. The better steps we are taking, and better solutions we are finding, the more careful we are going to be in finding a solution. In the beginning we allow random and bad steps to be performed, while later, we make better calculations.
In our algorithm we have ... KIE DO SOMETHING HERE

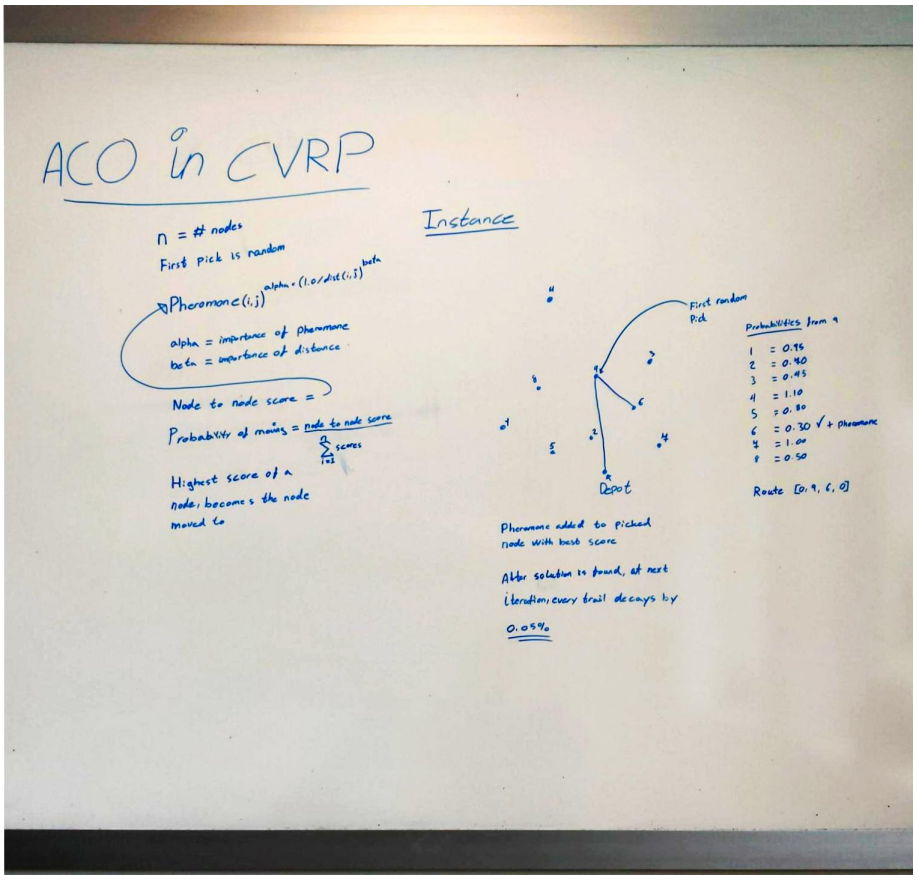
Algorithm 1 Metaheuristic - Simulated Annealing

3 Ant Colony Optimization

The Ant Colony Optimization (ACO) algorithm comes from the evolutionary algorithms, where computer science meets nature. In this algorithm, we have led us to be inspired by how ants find food in the nature. They are starting by spreading out in some area, randomly. When they seem to find trails, which could indicate to be good trails to find food, they leave pheromone behind them. The pheromone is used by other ants to determine their probability of taking a trail. If they are standing in a cross-section and they have to choose, they are taking the path with the highest pheromone. At some point in their exploration to find the best trail to obtain food, all the ants are using the same trails to get food and get back safe home.
We can apply the logic of the ants in the CVRP as well. By letting the instance the space of which the requests are "plotted" be the area to find a solution, we can start by sending one "ant" out to a point. From this point, we are going to calculate every probability of moving to the next point the one with the highest "score". We can calulate the probabilty of moving by the following:
First we establish the initial pheromone levels from all the points to every counter other point:

$$M = \begin{bmatrix} 0 & 0.50 & 0.20 \\ 0.125 & 0 & 0.60 \\ 1.20 & 0.355 & 0 \end{bmatrix}$$

Figure 1: Illustration of the ACO algorithm, showing how it can be used, and how it process next moves



4 BOXPLOT FIGURES

Appendices